■ ปราโมทย์ พลังสันติกุล

Visual Basic :

การตรวจสอบไวรัสอย่างง่าย ด้วยเทคนิค CRC

..วิธีการสำหรับตรวจสอบไวรัสอย่างง่าย ๆ ด้วยการใช้เทคนิค CRC (Cyclical Redundancy Check)...

ผมคิดว่าผู้ที่ใช้งานคอมพิวเตอร์แทบจะไม่มีใครที่ไม่รู้จักกับคำว่า *ไวรัสคอมพิวเตอร์* หรืออย่างน้อยอาจจะได้พบมาแล้วในเครื่องคอมพิวเตอร์ของคุณ และส่วนใหญ่คอมพิวเตอร์ของผู้ใช้งานก็มักจะมีโปรแกรมตรวจสอบไวรัสกันแทบทุกเครื่อง ซึ่งจะช่วยให้ผู้ใช้งานคอมพิวเตอร์ได้อุ่นใจขึ้นมาบ้างว่า อย่างน้อยเครื่องของเราก็ไม่มีทางที่จะติดไวรัสได้ ซึ่งเป็นการป้องกันปัญหาทางหนึ่ง แต่หากมีไวรัสชนิดใหม่เกิดขึ้นมา และบังเอิญโปรแกรมตรวจสอบไวรัสของเราไม่สามารถที่จะตรวจสอบไวรัสชนิดใหม่นี้ขึ้นมาได้ แน่นอนครับไฟล์ข้อมูลหรือไฟล์โปรแกรมของท่านก็จะต้องติดไวรัสได้ทันที และกว่าท่านจะรู้ ไวรัสก็ติดไฟล์ของท่านไปแล้ว ซึ่งหากเป็นไฟล์โปรแกรมก็คงไม่มีปัญหาเพียงหาโปรแกรมมาลงใหม่หากต้องการความแน่ใจ แต่ไฟล์ข้อมูลสิครับ หากไม่ได้สำรองไว้คงมีปัญหาไม่น้อย ซึ่งผู้ใช้งานบางท่านก็อาจจะหาโปรแกรมตรวจสอบไวรัสใหม่ยากกว่าผู้ที่อยู่ในสถานที่หาโปรแกรมรุ่นใหม่ได้ง่ายกว่าด้วยเหตุผลใดก็ตาม

เพราะฉะนั้นผมจึงได้นำวิธีการตรวจสอบไวรัสอย่างง่าย ๆ มานำเสนอในบทความนี้ ซึ่งจะเป็นประโยชน์ต่อการตรวจสอบไวรัสก่อน หากเรามีโปรแกรมตรวจสอบไวรัสซึ่งไม่ใช่รุ่นล่าสุดอยู่ หากเราจะตรวจสอบว่าไฟล์ของเราติดไวรัสหรือไม่ ก็จะมีวิธีการตรวจสอบไฟล์อยู่ 2 วิธี ที่ผมจะเลือกใช้ (เพราะผู้เขียนรู้จัก 2 วิธีนี้) คือ การตรวจสอบไฟล์ด้วย Checksum คือ การนำเอาจำนวนข้อมูลในไฟล์ทั้งหมดของแต่ละไบต์มาบวกกันจนได้ค่า ๆ หนึ่ง หรือไม่ก็ใช้วิธี CRC แต่ผมเลือกใช้วิธี CRC ครับ เพราะวิธีนี้มีความปลอดภัยมากกว่าวิธีของการ Checksum เพราะ

การคำนวณ CRC และเมื่อได้ค่า CRC แล้วเราจะไม่สามารถคำนวณค่ากลับมาได้ และโอกาสเกิดค่าซ้ำกันนั้นมีน้อยมากกว่าการใช้วิธี Checksum จึงเป็นเหตุผลที่เลือกใช้วิธีนี้

CRC (Cyclical Redundancy Check)

เมื่อพูดถึง CRC ผู้อ่านบางท่านอาจถามได้ว่า CRC คืออะไร CRC (Cyclical Redundancy Check) คือ สูตรที่ใช้ในการคำนวณหาค่าจากโพลิโนเมียล ซึ่งคิดขึ้นมาโดยองค์กร CCITT ซึ่งนำมาใช้กับโปรโตคอลในการสื่อสารข้อมูล เช่น ZMODEM เป็นต้น และการใช้ CRC ในการตรวจสอบข้อมูลในการสื่อสารเพื่อตรวจสอบความถูกต้องของข้อมูลได้รับความนิยมอย่างมากในปัจจุบัน

การคำนวณ CRC ได้จากวิธีการหารโพลิโนเมียล (Dividing polynomials) คือการหาร Message polynomial ด้วย generater polynomial โดยใช้หลักการ Modulo 2 ซึ่งก็คือการใช้ Exclusive-Or (XOR) เศษของการหารที่ได้ก็จะเป็นค่า CRC

ลิสติงที่ 1 ใช้สำหรับในการคำนวณหาค่า CRC

```
tblcrc32.dat
/* CRC-32 Table */
unsigned long crc32_table[256] =
{
 0x00000000, 0x77073096, 0xEE0E612C, 0x990951BA,
 0x076DC419, 0x706AF48F, 0xE963A535, 0x9E6495A3,
 0x0EDB8832, 0x79DCB8A4, 0xE0D5E91E, 0x97D2D988,
    :
    :
 0xBAD03605, 0xCDD70693, 0x54DE5729, 0x23D967BF,
 0xB3667A2E, 0xC4614AB8, 0x5D681B02, 0x2A6F2B94,
 0xB40BBE37, 0xC30C8EA1, 0x5A05DF1B, 0x2D02EF8D,
};
```

CRC-16

CRC-16 คือ เศษของโพลิโนเมียล ซึ่งอยู่บนฐานของ 16 บิตของข้อมูล ซึ่งต่างจาก CRC-CCITT ที่จะตั้งอยู่บนฐาน 8 บิตของข้อมูล และจำนวนหารโพลิโนเมียลสำหรับ CRC-16 จะใช้เช่นเดียวกับของ CCITT คือ ค่า $X^{16} + 1X^{15} + 1X^2 + 1$ และใช้ $X^{16} + 1X^{15} + 1X^2 + 1$ ในการคำนวณหาค่า CRC-16

CRC-32

เป็นการนำเอาการคิด CRC ทั้งของ CRC-16 และ CRC-CCITT มาใช้ร่วมกัน เพื่อนำมาใช้แก้ปัญหาข้อผิดพลาดที่เกิดจาก นัลบิต (NULL (ZERO) BIT) โดย CRC-32 จะแก้ปัญหาดังกล่าวด้วยการนำค่าของ CRC ที่ 0xFFFFFFFF มาใช้ในการแก้ปัญหาที่เกิดขึ้นกับ CRC-16 และ CRC-CCITT ส่วนสองวิธีคือ CRC-16 และ CRC-CCITT จะพิจารณาค่าศูนย์ตั้งแต่เริ่มแรก แต่ CRC-32 จะไม่ใช้ค่าศูนย์ในการเริ่มต้นของการคำนวณ CRC โดยใช้ค่า 0xFFFFFFFF ในการเริ่มการคำนวณ ซึ่งเราเรียกว่า การตรวจสอบเงื่อนไขแต่เริ่มแรก

การคำนวณ CRC-32 จะใช้แบบเดียวกับทั้งสองวิธีที่กล่าวข้างต้น คือ CRC-16 และ CRC-CCITT คือ การสร้างเศษของโพลิโนเมียล และส่วนของโพลิโนเมียลและใช้ $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ ในการคำนวณหาค่า CRC-32

โปรแกรมการคำนวณหาค่า CRC

หลังจากเราได้รู้หลักการของการคำนวณเพื่อการหาค่า CRC อย่างคร่าว ๆ แล้ว เราจะนำหลักการดังกล่าวมาใช้ในการหาค่า CRC ของไฟล์เพื่อทำการเก็บค่าไว้ใช้ในการตรวจสอบภายหลังได้ ซึ่งหากมีไวรัสเกิดกับไฟล์ของเราก็จะสามารถตรวจสอบได้ เนื่องจากจะมีการเปลี่ยนแปลงขนาดของ CRC เกิดขึ้นได้แม้ขนาดไฟล์ของเราจะไม่เปลี่ยนแปลงหรือเปลี่ยนแปลงน้อยมากจนเราไม่อาจสังเกตเห็นได้ก็ตาม

ในบทความนี้ผมจะแสดงการคำนวณหาค่า CRC เฉพาะ CRC-32 เท่านั้น เนื่องจากการคำนวณหาค่า CRC นั้นต้องการความรวดเร็วในการหาค่า และหากเราต้องคำนวณจำนวนข้อมูลที่มาก ๆ ก็จะทำให้เสียเวลาได้มาก ผมจึงได้พัฒนาส่วนของการคำนวณ CRC โดยการใช้ภาษา C ในการพัฒนาและสร้างเป็นไฟล์ Dynamic-Link Library (DLL) แทนการเขียนโค้ดด้วย Visual Basic แทน และอีกเหตุผลประการคือ เพื่อเป็นการยืดหยุ่นทำให้ผู้ที่สนใจสามารถนำไปใช้ได้กับทุกโปรแกรมที่สามารถใช้งานไฟล์ DLL ได้ ไม่ว่าจะเป็นโปรแกรมทางด้านฐานข้อมูล เช่น Visual Foxpro, Visual dBase เป็นต้น นอกจากนี้ยังได้แสดงส่วนของโค้ดที่เป็นภาษา C ซึ่งไม่ได้พัฒนาเป็นไฟล์ Dynamic-Link Library (DLL) สำหรับผู้สนใจที่ไม่ต้องการใช้งานในส่วนของ DLL ก็สามารถที่จะใช้งานได้ที่ระบบดอส (DOS) ได้เช่นเดียวกันครับ

ลิสติงที่ 2 โปรแกรมคำนวณหาค่าของตารางที่ใช้ในการคำนวณหาค่า CRC

CRCTABLE.C

```
/* Filename: CRCTABLE.C
:
Note : show to screen use c:\>crctable
: write to file use c:\>crctable > filename
: Ex c:\>crctable > tblcrc32.dat
*/

#include <stdio.h>

#define POLYNOMIAL_CRC32 0xEDB88320L

unsigned long table[256];
unsigned long CalculateCRC32Table(int);

void main(void)
{
 unsigned int i;

 for(i=0;i<256;i++)
 {
 table[i] = CalculateCRC32Table(i);
 }

 printf("/* CRC-32 Table */\n");
 printf("unsigned long crc32_table[256] = \n{");

 for(i=0;i<256;i++)
 {
 if((i%4) == 0)
 printf("\n ");

 printf("0x%08lX",(unsigned long) table[i]);

      if(i!=256)
 printf(", ");
 }
 printf("\n};\n");

}

unsigned long CalculateCRC32Table(int i)
{
 int j;
 unsigned long crc;
 crc = i;

 for(j=8;j>0;j--)
 {
 if(crc&1)
 crc = (crc>>1) ^ POLYNOMIAL_CRC32;
 else
 crc >>=1;
 }
 return(crc);

}
```

ลิสติงที่ 3 โปรแกรมคำนวณหาค่า CRC ของไฟล์

DOSCRC32.C

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include "tblcrc32.dat"

#define LOCATES_MEMORY 32766

long CalculateCRC32(char *FileNameCRC)
{
 FILE *FileChkCRC;
 unsigned char *buffer;
 unsigned int i,j;
 unsigned long tmp1, tmp2;
 unsigned long crc32;

 if((FileChkCRC=fopen(FileNameCRC,"rb")) == NULL)
 {
      return(1);              //error: no file open
 }
 if((buffer = (unsigned char *) malloc(LOCATES_MEMORY))
== NULL)
 {
 return(2);                   //error: out of memory
 }

 crc32 = 0xFFFFFFFF;

 while(1)
 {
 i = fread(buffer, 1, LOCATES_MEMORY, FileChkCRC);
 if(i==0)
 {
 if(feof(FileChkCRC))
 {
 crc32 = (~crc32) - 1;
          return(crc32);
      }
      else
 {
 continue;
 }
 }

 for(j=0;j<i;j++)
 {
 tmp1 = (( crc32 >> 8 ) & 0x00FFFFFFL);
      tmp2 = crc32_table[(crc32^ buffer[j]) & 0x000000FFL];
      crc32 = tmp1^tmp2;
 }
 }
 }

 void main(int argc, char *argv[])
 {
 long crc32=0;

 switch(argc)
 {
 case 1:
 printf("Usage : Doscrc32 filename \n");
 printf(" : filename = File Check CRC \n");
 break;
 case 2:
 crc32 = CalculateCRC32(argv[1]);
 if(crc32 == 1)
 {
 printf("Error : no such file");
 return;
 }
 if(crc32 == 2)
 {
 printf("Error : out of memory");
 return;
 }
 printf("/* Calculate CRC-32 */\n");
 printf("Crc32 = %08lX\n",crc32);
 break;
 }
 }
```

ลิสติงที่ 4
โปรแกรมคำนวณหาค่า CRC ของไฟล์ ในแบบ DLL

```
extern "C" long WINAPI CalculateCRC32(char *FileNameCRC,
unsigned long *CRC32)
{
 FILE *FileChkCRC;
 unsigned char *buffer;
     unsigned int i,j;
     unsigned long tmp1, tmp2;
     unsigned long crc32;

     if((FileChkCRC=fopen(FileNameCRC,"rb")) == NULL)
     {
          return(1);          //error: no file open
     }

     if((buffer = (unsigned char *) malloc(LOCATES_MEMORY))
     == NULL)
     {
          return(2);          //error: out of memory
     }

     crc32 = 0xFFFFFFFF;

     while(1)
     {
          i = fread(buffer, 1, LOCATES_MEMORY, FileChkCRC);
          if(i==0)
          {
                    if(feof(FileChkCRC))
                    {
                              *CRC32 = (~ crc32) - 1;
                              return(0);
                    }
                    else
                    {
                              continue;
                    }
          }
          for(j=0;j<i;j++)
          {
tmp1 = (( crc32 >> 8 ) & 0x00FFFFFFL);
tmp2 = crc32_table[(crc32^ buffer[j]) & 0x000000FFL];
crc32 = tmp1^tmp2;
}
}
}
```

ผมขอไม่อธิบายโค้ดการทำงานในส่วนของภาษา C ทั้งบนระบบดอสและส่วนของ Dynamic-Link Library (DLL) คงเป็นงานของท่านผู้อ่านจะศึกษาในส่วนโปรแกรมที่เป็นภาษา C แต่ผมจะนำเสนอในส่วนของการนำไปใช้งานแทน เพราะเราได้ทำการพัฒนาเป็นไฟล์ Dynamic-Link Library (DLL) แล้ว

อธิบายการทำงานของโพรซีเดอร์ Calculate CRC32 DLL

โปรแกรม Visual Basic : Calculate CRC32 DLL ในการใช้งาน WINCRC32.DLL ที่สำคัญมีดังนี้

FrmChkCRC32.FRM

```
Option Explicit  กำหนดให้มีการประกาศการใช้ตัวแปรทุกครั้ง
การประกาศค่าฟังก์ชัน API เพื่อใช้งาน WINCRC32.DLL
คือ ฟังก์ชันในการคำนวณค่า CRC32
Private Declare Function CalculateCRC32 Lib
"WINCRC32.DLL" (ByVal inputfile As String,
ByRef CRCResult As Long) As Long
Dim strFileNameChkCRC As String
' ตัวแปรใช้ในการกำหนดชื่อไฟล์ที่ต้องการคำนวณหาค่า CRC
Dim strCRC32 As String
' ตัวแปรใช้ในการเก็บค่า CRC ที่คำนวณได้
```

Private Sub cmdCalcCRC32_Click()

โพรซีเดอร์ cmdCalcCRC32_Click() ใช้ในการคำนวณหาค่า CRC ของไฟล์ที่เราได้ทำการเลือกจากลิสต์ไฟล์ โดยชื่อไฟล์จะส่งผ่านไปในฟังก์ชัน CalcCRC32 (strFileNameChkCRC, ChkCRC) ใน strFileName ChkCRC และฟังก์ชันดังกล่าวจะส่งค่ากลับมา โดยส่งค่า CRC กลับมาที่ตัวแปร ChkCRC และค่า Error Code ว่าเกิดการผิดพลาดในระหว่างการทำงานหรือไม่

```
RES = CalculateCRC32(strFileNameChkCRC, ChkCRC)
lblChkCRC32 = "CRC32 : " & Hex(ChkCRC)
If RES = 0 Then lblErrCRC32 = "Error Msg : Non"
If RES = 1 Then lblErrCRC32 = "Error Msg : No file open"
If RES = 2 Then lblErrCRC32 = "Error Msg : Out of memory"
```

หากค่า RES มีค่าเท่ากับศูนย์ (RES = 0) โปรแกรมสามารถทำการคำนวณหาค่า CRC ได้เป็นผลสำเร็จ แต่หากส่งค่ากลับมาเป็นหนึ่ง (RES = 1) แสดงว่าไม่สามารถเปิดไฟล์ที่ต้องการคำนวณหาค่า CRC ได้ และหากส่งค่ากลับมาเป็นสอง (RES = 2) แสดงว่าไม่สามารถจองหน่วยความจำเพื่อใช้ในการคำนวณค่า CRC

Private Sub cmdSaveFNameCRC_Click()

โพรซีเดอร์ cmdSaveFNameCRC_Click() เป็นโพรซีเดอร์ที่ใช้ในบันทึกการคำนวณค่า CRC เพื่อเก็บลงไฟล์เพื่อใช้ในการอ้างอิงภายหลัง

```
FileNumber = FreeFile ' ส่งค่ากลับของหมายเลขไฟล์
ที่ยังไม่ได้ถูกเปิดใช้งาน
strFileName = txtFileNameSaveCRC
Open strFileName For Append As #FileNumber
' Create filename. ' เปิดไฟล์เพื่อเขียน
Write #FileNumber, strFileNameChkCRC, strCRC32
' เขียนลงไฟล์
Close #FileNumber ' Close file.
' ปิดไฟล์
```

Private Sub Drive1_Change()

โพรซีเดอร์ Drive1_Change() ใช้ในการเปลี่ยนไดรฟ์ใหม่ตามที่ต้องการ

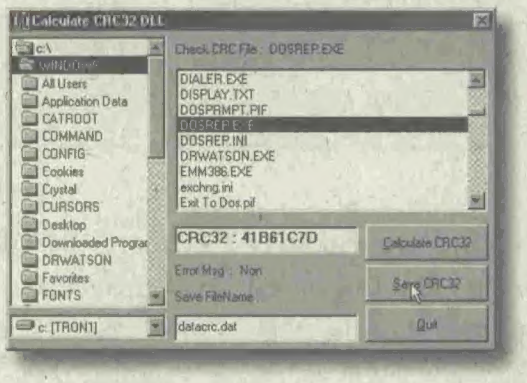
Private Sub File1_Click()

โพรซีเดอร์ File1_Click () ใช้สำหรับเลือกไฟล์ที่ต้องการคำนวณหาค่า CRC ซึ่งจะใช้กำหนดค่าใน strFileNameChkCRC

```
strFileNameChkCRC = File1.Path & "\" & File1.filename
lblFileName.Caption = "Check CRC File : " &
    UCase(File1.filename)
```

Private Sub Dir1_Change()

โพรซีเดอร์ Dir1_Change () ใช้กำหนดพาธไฟล์ที่กำหนด



รูปที่ 1 โปรแกรม Calculate CRC32 DLL

การใช้งานโปรแกรม Calculate CRC32 DLL

โปรแกรมตัวอย่าง Calculate CRC32 DLL แสดงดังรูปที่ 1 ขั้นตอนการใช้งานก็เพียงแค่เลือกไฟล์ที่ต้องการคำนวณหาค่า CRC จากไฟล์ลิสต์บ็อกซ์ และเลือกปุ่ม Calculate CRC32 โปรแกรมก็จะทำการคำนวณหาค่า CRC และจะแสดงค่า CRC ของไฟล์นั้น ๆ ดังรูปที่ 1

หมายเหตุ ต้องคัดลอกไฟล์ WINCRC32.DLL ไปไว้ที่ไดเรคทอรีวินโดว์ในซับไดเรคทอรีซิสเต็มส์ก่อน

ท้ายบทความ

หลังจากได้ทดสอบการใช้งานโปรแกรมตัวอย่างเพื่อคำนวณหาค่า CRC แล้ว ยังมีบางส่วนที่ยังไม่สมบูรณ์ คือ การคำนวณหาค่า CRC ทีเดียวทั้งไดเรคทอรีเลยเพื่อความรวดเร็วหรือเลือกได้ตามที่ต้องการ และการนำค่าของ CRC ที่ได้ทำการบันทึกไว้แล้วกลับมาเปรียบเทียบกับการหาค่า CRC ครั้งใหม่ว่ามีการเปลี่ยนแปลงหรือไม่ คงต้องฝากไว้ให้ท่านผู้อ่านนำไปพัฒนาต่อครับ ผมหวังว่าบทความนี้คงพอจะเป็นแนวทางให้ท่านผู้อ่านนำไปประยุกต์ใช้ในด้านต่าง ๆ ได้มากขึ้น นอกจากการใช้ตรวจสอบไวรัสแล้ว อย่าลืมเขียนมาเล่าสู่กันฟังบ้างนะครับ ที่ VLinux@hotmail.com

จากการใช้หลักการ CRC (Cyclical Redundancy Check) ในการคำนวณหาค่า CRC ของไฟล์ ผมคิดว่าจะมีประโยชน์อย่างมาก เพราะปัจจุบันไวรัสมีมากมายและเกิดขึ้นมาใหม่เป็นจำนวนมาก ซึ่งหากจะรอแต่โปรแกรมตรวจสอบไวรัส ท่านอาจจะพบไวรัสก่อนที่จะมีโปรแกรมกำจัดไวรัส แต่หากใช้วิธีป้องกันด้วยหลักการ การหาค่า CRC เป็นการป้องกันทางหนึ่งก่อน ก็จะทำให้เรารู้ได้ว่าข้อมูลและโปรแกรมของเราได้ติดไวรัสแล้วหรือยัง ซึ่งเป็นการป้องกันไม่ให้ไวรัสแพร่ออกไปมากกว่าที่เป็น และยังเป็นการลดความเสียหายลงได้อันเนื่องมาจากไวรัส จากนั้นแล้วค่อยหาทางแก้ไข ในขั้นตอนต่อไป "การป้องกันไว้ก่อนเป็นหนทางที่ดีกว่าการมาแก้ไขภายหลัง" ส่วนรายละเอียดเกี่ยวกับ CRC นั้นผมได้ความคิดมาจากหนังสืออ้างอิงเล่มที่ 1 ประกอบกับเล่มที่ 2 ครับ

หากสนใจต้องการซอร์สโค้ดก็สามารถติดต่อขอใช้บริการได้จากซอฟต์เมล์ ฉบับนี้ก็ขอจบเพียงแค่นี้แล้ว พบกันใหม่ สวัสดีครับ

หนังสืออ้างอิงเกี่ยวกับ CRC

1. Ken Miller. Building DLLs Visual Basic Programmer's Journal September 1996.
2. Practical Algorithms for Programmers by Andrew Binstock and John Rex (ISBN 0-201-63208-X)
Itseltbase on an article by Mark R. Nelson in the May 1992 issue of Dr.Dobb.s Journal
3. Tim Kientzle. The Working Programmer's Guide to Serial Protocals. Coriolis Group Books